A Creativity Survey of Distributed Database System

JIAHAO FU

Institute of Computing Technology, Chinese Academy of Sciences; fujiahao211@mails.ucas.ac.cn

WEI LI

Institute of Computing Technology, Chinese Academy of Sciences; liwei@ict.ac.cn

Abstract: *Distributed database system are widely used because of the rapid development of the Internet. With the ever-increasing demand, the boost performance and minimize resource and data contention are taken into consideration. A great distributed physical design, which determines where to place data, and which data item to replicate and partition, would help. This paper classification the development of physical design based on Michael's work and its references in research problems, research methods and measurement methods. Finally we put forward some suggestions for future research.*

Keywords: *Distributed Database System*

1 INTRODUCTION

With the traditional database technology becoming mature, the rapid development of computer network technology and the expansion of application range, database application has been widely established on the computer network. At this time, the centralized database system shows its shortcomings: the data has been distributed and stored on the network according to the actual needs, and then the centralized processing will inevitably cause the communication overhead; The application program is concentrated on a computer, once the computer breaks down, the whole system is affected, the reliability is not high; The scale and configuration of the system are not flexible enough, and the scalability of the system is poor. In this situation, the research and development of the database system with the main characteristics of distributed has attracted people's attention. Distributed database is the combination of database technology and network technology, which has formed a branch in database field. Distributed database systems store and manage large amounts of data, copy and partition data, and distribute its transactions across multiple nodes. The data replication and partitioning solution selects a distributed database from a physical design, which can boost performance and minimize resource and data contention. We did a creativity survey of distributed database system for induction and summary and try to find future research.

The rest of the paper is organized as follows. Section II gives the classification of research objects of Database System. Section III introduces the classification of research methods. Section IV introduces the comparison of experimental analysis in related literature. Section V discusses the research opportunities in future work and Section VI concludes the paper.

2 CLASSIFICATION OF RESEARCH OBJECTS

Table 1: Different Research Objects

Database System	Management of Data			
	Partition	Replication	Master	Index
Distributed	I. [1][4][7][9][11] [13][14][15][16][17]	II. [1][3][4][5][6] [8][10][12][18][19]	III. [1][2][3]	IV. [11]
Centralized	V. [11]	VI. [5][6]	VII.	VIII. [11]

2.1 Criteria

The distributed database systems’ physical design has great related with data control. Besides, during the management of data there still some works would be done in centralized database systems. Therefore, in this section, two independent and different criteria would be used to divide research objects into different types:

1) **Database System**. There are two types here: **Distributed Data System** or **Centralized Database System**. This is the classification of database system in maximum dimensions. Some methods and technologies used in centralized database system always can used in distributed data system, so some works are intersectional.

2) **Management of Data**. There are four kinds of data management here: **Partition, Replication, Master** or **Index**. In distributed data system those four kinds of data management are readily comprehensible: partition means breaking the data and store them in different database, replication means some data can be stored in some database simultaneously, while there are some data conflicting or not, we need have an authoritative data management that means master, finally the index is just for quicker data selecting. Those four kinds of data management can be used for data backup and recovery in centralized database system or other uses.

2.2 The Classification

Based on the appeal classification standard, we give the classification in Table 1. The meaning of each class is as follows:

Type I: This type is data partition in distributed database system.

Type II: This type is data replication in distributed database system.

Type III: This type is the data master in distributed database system.

Type IV: This type is the data index in distributed database system.

Type V: This type is data partition in centralized database system.

Type VI: This type is data replication in centralized database system.

Type VII: This type is the data master in centralized database system.

Type VIII: This type is the data index in centralized database system.

2.3 Explanation of Different Types

References ([4][7][9][13][14][15][16][17]) belong to Type I. Reference [4] use workload to decide the replication. Reference [7] focus on the fine-grained configuration for the partitioned main memory databases. Reference [9] take distributed joins into consideration. Reference [13] is based on the partially replicated database system and it focus on query centric partitioning. Reference [14] proposes a partitioning for general database schemas. Reference [15] is for ad-hoc query workloads. Reference [16] is for distributed transaction processing systems. Reference [17] take shared-everything OLTP into consideration.

References ([3][4][8][10][12][18][19]) belong to Type II. Reference [3] is based on partitioned snapshot isolation database system. Reference [4] focus on replication and partition based on workload. Reference [8] claims the dangers of replication and have a solution. Reference [10] need to considerate the scaling transaction. Reference [12] is based on partially replicated databases. Reference [18] proposes a low bound data replication algorithm. Reference [19] update everywhere database replication approach based on snapshot isolation within the open-source database system PostgreSQL.

References ([2][3]) belong to Type III. Reference [2] proposes the DynaMast, which is benefits from these advantages by dynamically transferring the mastership of data, or remastering, among sites using a lightweight metadata-based protocol. Reference [3] proposes a multi-master replication.

References ([1]) belong to Type I, II, III. Reference [1] uses the history workload to train a cost model, according this learned cost model to make a compound decision on partition, replication and mastering.

References ([11]) belong to Type I, IV, V, VIII. Reference [11] focus on database's partition and index, this paper proposes an optimization method. The query time will be optimized according to the user's query history pattern. Only virtual partition is used to access raw data, and linear programming and greedy algorithm are used to optimize the cost model. Through this model, we can get an optimized result.

References ([5][6]) belong to Type II, VI. Reference [5] proposes an algorithm for lazy database replication. The algorithm would guarantee the ordering. Reference [6] is another algorithm for lazy database replication, the difference from last one is this algorithm guarantees snapshot isolation rather than ordering.

No references belong to Type VII. In a centralized database system, same data in different database are considered a backup, so there would not have data conflict. What we need to do is enable backup databases when fatal errors occur to the database. Therefore, very few studies have done this.

3 CLASSIFICATION OF RESEARCH METHODS

Table 2. Different Research Methods

Method of Configuration	Using Machine Learning Model？	
	Yes	No
Dynamic	I. [1]	II. [2][4][7][9][11][13][14][15][16][17][18]
Static	III.	IV.[3][5][6][8][10][12][19]

3.1 Criteria

We need to configure the setting before we run a database system. And some time we can use Machine Learning to generate greater setting for us. Therefore, in this section, two independent and different criteria would be used to divide research objects into different types:

1) **Using Machine Learning Model**. There are two types here: **Yes** or **No**. With the rapid development of artificial intelligence, people can get great idea from machine learning model sometimes. Therefore, we can use the model to help us to get greater parameters.

2) **Method of Configuration**. There are two kinds of configuration here: **Dynamic** or **Static**. As we can know, static configuration is easy, but the system is not static. We can change the configuration to adapt the system more excellent by using dynamic configurate.

3.2 The Classification

Based on the appeal classification standard, we give the classification in Table 2. The meaning of each class is as follows:

Type I: This type is using machine learning model to get setting parameter and dynamic configurate.

Type II: This type is get setting parameter without machine learning model and dynamic configurate.

Type III: This type is using machine learning model to get setting parameter and static configurate.

Type IV: This type is get setting parameter without machine learning model and static configurate.

3.3 Explanation of Different Types

References ([1]) belong to Type I. Reference [1] uses linear regression models consume a vector of inputs and output a scalar prediction of operations' latency.

References ([2][4][7][9][11][13][14][15][16][17][18]) belong to Type II. Reference [2] is for dynamic mastering which is adaptive and support multi-mastering. Reference [4] proposes a strategy based on graph, which is organized with database and workload. Reference [7] takes fine-grained live reconfiguration. while reconfiguration, there is no need to shut down the server. Reference [9] proposes an adaptive partitioning to reduce distributed joins' cost. In reference [11], the query time will be optimized according to the user's query history pattern. Only virtual partition is used to access raw data, and linear programming and greedy algorithm are used to optimize the cost model. Reference [13] proposes an analytical model that is workload aware and scalable, which uses linear programming algorithms to solve optimization results. Reference [14] using elastic algorithm based on Heat Graph. Reference [15] creates a partition tree and maintain the partition tree according to the user's query to achieve an adaptive and robust partition scheme. In reference [16], an elastic partitioning framework, E-Store framework, is proposed, which includes E-Monitor and E-Planner for hot spot tracking and heuristic data movement planning respectively. Reference [17] using logical partitioning and MBR-Tree, which is also based on workload. Reference [18] provide a lower bound on the performance of any dynamic replication algorithm.

No references belong to Type III.

References ([3][5][6][8][10][12][19]) belong to Type IV. Those references are all downplay database configuration or unfocused it.

4 REVIEW OF EXPERIMENTAL ANALYSIS

In this section, we will classify the metric of evaluation and system parameters, as shown in Table 3. In Table 3, all experimental analysis is also classified according to the metric and parameters. It can be seen from Table 3 that most of the references compare throughput, delay, and response time.

Table 3. Experiments with Different Metric and Parameters

Parameters	Metric			
	Throughput	**Delay**	**Response time**	**Other**
Hardware	I. [1][3][5][6][10] [12][13][17]	II. [12]	III. [3][5][6][12][13]	IV. [10][13][17] [18]
Software	V.[1][2][3][4] [5][6][7][10][12] [13][14][16][17][19]	VI. [1][2][7][10] [12][14][16]	VII.[3][5][6][12][13] [19]	VIII.[4][8][9] [11][13][14][15] [17][18][19]

4.1 Metric of Evaluation

Throughput means the number of database transactions per unit of time. The formula is as follows:

$$\text{Throughput} = \frac{\text{Total Transaction Number}}{Time}$$

Delay means request response time in and out of the database system.

Response time means the client request starts until it is time to receive a response.

Other metric includes Network delay, Distributed transaction ratio, Cost and so on.

4.2 System Parameters

Hardware represents the hardware environment where experiment establishes. Such as the network topology, number of clients or servers and so on.

Software represents experimental software configuration. Such as configuration variables, algorithm parameters or experimental data and so on.

4.3 Experimental Comparison

In reference [1], the author compares the throughput and transaction delay of database system in different number of clients. And the model accuracy and model cost are evaluated.

In reference [2], the author compares throughput and transaction delay in different number of clients.

In reference [3], the author compares throughput, additional delay and read-only transactions' response time.

In reference [4], the author compares distributed transaction ratio for partition number and graph sizes.

In reference [5][6], the author compares throughput, read-only transaction and read-write transactions' response time for different number of clients.

In reference [7], the author compares throughput and delay during the reconfiguration and after reconfiguration.

In reference [8], there is no experiment.

In reference [9], the author compares running time and buffer sizes for different data sizes.

In reference [10], the author compares throughput and the algorithm's state transit cost.

In reference [11], the author compares the cost of memory and query time for different query.

In reference [12], the author compares response time, the ratio of transaction commit and local transaction ratio.

In reference [13], the author compares throughput and average response time in different number of servers, requests or back-end and so on.

In reference [14], the author compares throughput, delay and distributed transaction ratio.

In reference [15], the author compares upfront overhead and first query runtime.

In reference [16], the author compares throughput and delay using variety of algorithms.

In reference [17], the author compares throughput, the time of transaction execute and the time and space overhead of the algorithm.

In reference [18], the author compares average communication cost savings in different read and write modes.

In reference [19], the author compares response time, throughput, the number of replicas and the cost of replicating.

5 DISCUSSION AND SUGGESTION

This paper discusses the research methods and research objects of various references and finds that there are still some studies in distributed database system that have not been paid attention to by researchers. Therefore, this paper puts forward the following directions, which can provide directions for future distributed database system research:

1) Using Machine Learning model to configure settings statically. Artificial intelligence is developing so fast that many people can do things that they can't make up their minds quickly. Using machine learning models to train a static configuration as the initial state of the system is a good research area.

2) The index of the database directly affects the execution speed of the database in the query transaction, but whether the original index can still be used after the database performs large number of update transactions, and whether the dynamic index modification can speed up the query is also a topic worth studying.

6 CONCLUSIONS

Distributed database systems have a lot of room for improvement in the future. And we did a creativity survey of distributed database system based on M. Abebe's work and its reference. We classify the research objects, research methods and metrics into different classification. Some of these areas have been extensively studied, However, there are still some studies in distributed database system that have not been paid attention to by researchers or have little research value. Finally, after the classification, we also show the research directions with better research prospects.

References

[1] M. Abebe, B. Glasbergen, and K. Daudjee, MorphoSys: automatic physical design metamorphosis for distributed database systems, VLDB, 2021.

[2] M. Abebe, B. Glasbergen, and K. Daudjee. DynaMast: Adaptive dynamic mastering for replicated systems. In IEEE 36th International Conference on Data Engineering (ICDE), pages 1381–1392. IEEE, 2020.

[3] P. Chairunnanda, K. Daudjee, and M. T. Ozsu. Confluxdb: Multi-master replication for partitioned snapshot isolation databases. PVLDB, 7(11):948–958, 2014.

[4] C. Curino, E. Jones, Y. Zhang, and S. Madden. Schism: a workload-driven approach to database replication and partitioning. PVLDB, 3(1-2):48–57, 2010.

[5] K. Daudjee and K. Salem. Lazy database replication with ordering guarantees. In IEEE 20th International Conference on Data Engineering (ICDE), pages 424–435. IEEE, 2004.

[6] K. Daudjee and K. Salem. Lazy database replication with snapshot isolation. In Proceedings of the 32nd International Conference on Very Large Data Bases (VLDB), pages 715–726, 2006.

[7] A. J. Elmore, V. Arora, R. Taft, A. Pavlo, D. Agrawal, and A. El Abbadi. Squall: Fine-grained live reconfiguration for partitioned main memory databases. In Proceedings of the 2015 ACM International Conference on Management of Data (SIGMOD), pages 299–313. ACM, 2015.

[8] J. Gray, P. Helland, P. O'Neil, and D. Shasha. The dangers of replication and a solution. ACM SIGMOD Record, 25(2):173–182, 1996.

[9] Y. Lu, A. Shanbhag, A. Jindal, and S. Madden. Adaptdb: adaptive partitioning for distributed joins. PVLDB, 10(5):589–600, 2017.

[10] Y. Lu, X. Yu, and S. Madden. Star: Scaling transactions through asymmetric replication. PVLDB, 12(11):1316–1329, 2019.

[11] M. Olma, M. Karpathiotakis, I. Alagiannis, M. Athanassoulis, and A. Ailamaki. Slalom: Coasting through raw data via adaptive partitioning and indexing. PVLDB, 10(10):1106–1117, 2017.

[12] V. Padhye, G. Rajappan, and A. Tripathi. Transaction management using causal snapshot isolation in partially replicated databases. In IEEE 33rd International Symposium on Reliable Distributed Systems (SRDS), pages 105–114. IEEE, 2014.

[13] T. Rabl and H.-A. Jacobsen. Query centric partitioning and allocation for partially replicated database systems. In Proceedings of the 2017 ACM International Conference on Management of Data (SIGMOD), pages 315–330. ACM, 2017.

[14] M. Serafini, R. Taft, A. J. Elmore, A. Pavlo, A. Aboulnaga, and M. Stonebraker. Clay: fine-grained adaptive partitioning for general database schemas. PVLDB, 10(4):445–456, 2016.

[15] A. Shanbhag, A. Jindal, S. Madden, J. Quiane, and A. J. Elmore. A robust partitioning scheme for ad-hoc query workloads. In Proceedings of the 2017 Symposium on Cloud Computing (SoCC), pages 229–241. ACM, 2017.

[16] R. Taft, E. Mansour, M. Serafini, J. Duggan, A. J. Elmore, A. Aboulnaga, A. Pavlo, and M. Stonebraker. E-store: Fine-grained elastic partitioning for distributed transaction processing systems. PVLDB, 8(3):245–256, 2014.

[17] P. T¨oz¨un, I. Pandis, R. Johnson, and A. Ailamaki. Scalable and dynamically balanced shared-everything oltp with physiological partitioning. The VLDB JournalThe International Journal on Very Large Data Bases (VLDBJ), 22(2):151–175, 2013.

[18] O. Wolfson, S. Jajodia, and Y. Huang. An adaptive data replication algorithm. ACM Transactions on Database Systems (TODS), 22(2):255–314, 1997.

[19] S. Wu and B. Kemme. Postgres-r (si): Combining replica control with concurrency control based on snapshot isolation. In IEEE 21st International Conference on Data Engineering (ICDE), pages 422–433. IEEE, 2005.